# Блоки инициализации

Просто посреди ничего пишем

{

System.out.print(“Write your text”)//Код самого блока

}

Бывают еще и статические блоки

static {

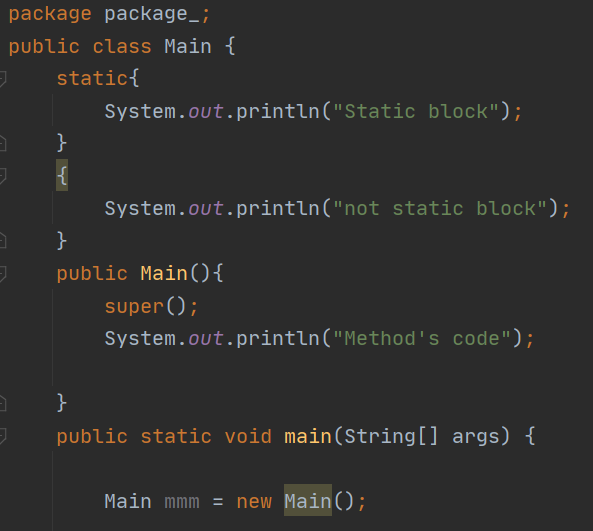
//другая реализация

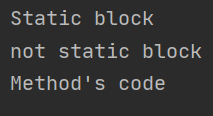
}

Статический блок запускается при первом обращении к классу, обычный при создании экземпляра класса.

При создании объекта вывод идет в следующем порядке:

* Статический блок в родительском классе
* Статический блок в наследнике
* Super (даже если его не прописать, он вызывается не явно)\*
* Не статический блок
* Код внутри конструктора



\*я так понял всегда при вызове конструктора по умолчанию тригериться конструктор по умолчанию суперкласса

Статический блок срабатывает и без создания объекта (например, мы можем обратиться к статическому методу класса, и тогда, перед методом выполниться статик блок). То есть любое первое обращение к классу = срабатывание статик блок.

Статических и обычных блоков инициализации может быть много, и они выполняются по очереди

Некоторые шизики называют нестатический блок инициализации анонимным блоком

Могу впишнуть блок и внутрь конструктора (мб это и не надо, а чел для примера показал)

Сначала запускаются статические блоки из супер.

Самый полный порядок вызова:

1. Статические поля базового класса
2. Статический блок инициализации данного класса (если в коде мы напишем сначала блок, который ссылается на поле, и только потом его объявим, то получим в лицо ошибку, но в принципе, можно в обратном порядке поставить)
3. Статические поля наследника
4. Статические блоки наследника
5. Поля (глобальные переменные) базового класса (??????)
6. Нестатический блок базового класса
7. Пустой конструктор базового класса (а может на пункт раньше)
8. Поля (глобальные переменные) наследника
9. Нестатический блок производного класса
10. Конструктор наследника

# Области видимости

У нас в коде есть много скобок {}, для них работает общее правило:

То, что находиться внутри фигурных скобок имеет доступ ко всему, что находиться во вне, но не имеет доступа ко всему, что находиться внутри других фигурных скобок, которые находятся внутри данных.

На самом деле я могу просто так написать внутри метода

{

Код

}

Просто чтобы изолировать область видимости (и это не будет являться блоком инициализации. Чел не рекомендует этого делать, а, вместо этого, выносить то, что ты хочешь изолировать в отдельный метод.